Computing controlled invariant sets in two moves

Tzanis Anevlavis and Paulo Tabuada

Abstract— In this paper we revisit the problem of computing controlled invariant sets for controllable discrete-time linear systems. We propose a novel algorithm that does not rely on iterative computations. Instead, controlled invariant sets are computed in two moves: 1) we lift the problem to a higher dimensional space where a controlled invariant set is computed in closed-form; 2) we project the resulting set back to the original domain to obtain the desired controlled invariant set. One of the advantages of the proposed method is the ability to handle larger systems.

I. INTRODUCTION

A controlled invariant set for a control system is, intuitively, a set with the following property: any trajectory starting in this set can be forced to remain inside it by a suitable choice of inputs. Such sets are useful since they can express "safety" properties in the sense that staying always within these sets ensures that something bad never happens. Controlled invariant sets play an important role in several control design problems, e.g., they are used as safe sets in constrained control, and they guarantee feasibility of optimization problems arising in model predictive control [14]. Moreover, their computation is essential to solve safety problems: if a trajectory is to remain forever in a set of safe states, then it must start in a controlled invariant set within that set.

Invariant sets for control systems have a long history in control; beginning with the pioneering work of [1], [2] on their computation, many other contributions followed and are documented in [3], [4], and more recently with their central role in the synthesis of correct-by-design systems [18], [20], [23].

Subsequently a substantial effort has been devoted to computing controlled invariant sets for continuous-time, discrete-time and hybrid systems. Many well-known approaches are based on the famous iterative procedure [1], [2], [7], see Section 2 for details. That procedure is proven to yield the maximal controlled invariant set upon termination. However, computing a controlled invariant set is still extremely challenging. For many important classes of systems finite termination is not guaranteed and other stopping criteria and relaxations are considered [4]. Moreover, the sets obtained become more complex with each iteration, consequently making solutions by this procedure intractable for high dimensional systems.

Attempts to overcome the difficulties and restrictions of computing controlled invariant sets relied on solving optimization problems, mostly linear programs, and using approximation techniques. Unavoidably, these techniques suffer from the efficiency-accuracy tradeoff, and the curse of dimensionality [8], [12], [24]. Recently, in [15], the authors presented an efficient method using semidefinite programming. Other attempts utilize sampling-based approximations [9] for linear sampled-data systems to keep the system within a set over some small specified time horizon. They are able to handle larger systems, but only provide invariance on finite time intervals.

In some cases, it is possible to compute the maximal controlled invariant set in finitely many steps [4], [13]. An important class of systems that ensures finite termination are discrete-time linear systems: if the system is controllable, and the constraints on states and controls are given by a finite union of hyper-rectangles, then the maximal control invariant set is finitely determined, see [20], [23], [25]. For another popular class of systems, namely the controllable discrete-time linear systems with bounded perturbations, where the state and control constraints are assumed to be polytopes, an approach to compute *robust* controlled invariant sets is given in [19].

In this paper we address the problem of computing a controlled invariant subset of a convex and compact polyhedral set for the class of controllable, discrete-time linear systems. We propose a novel algorithm that computes controlled invariant sets in only two moves: 1) it lifts the set whose controlled invariant subset we wish to compute to an hyper-rectangle in a higher dimensional space where its Maximal Controlled Invariant Subset (MCIS) can be computed in closed-form; 2) it projects this representation back to the original domain, and therefore is free of many of the numerical discrepancies or trade-offs of other state-of-the-art methods. Although the proposed algorithm does not compute the MCIS, it is nevertheless complete in the sense that if the MCIS is non-empty it will compute a controlled invariant set.

The paper is organized as follows, in Section II the formal problem setup is presented, along with the basic definitions and notations. Next, Section III illustrates our method in detail, and provides proofs of correctness and completeness. Finally, in Section IV we provide a number of examples showing the computational efficiency of our method that handles arbitrarily high dimensional systems, before concluding our remarks in Section V.

II. PROBLEM SETUP AND BASIC DEFINITIONS

In this section, the problem of computing a controlled invariant subset $\mathcal{D}$ of a compact polyhedron $D \subset \mathbb{R}^n$ given a linear discrete-time system Σ is formalized.

Problem 1: Consider a discrete-time linear system Σ defined by (A, B):

$$x_{k+1} = Ax_k + Bu_k, \tag{1}$$

where $x_0 \in D, u_k \in \mathbb{R}, A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^n$, and $D \subset \mathbb{R}^n$ is a compact polyhedron:

$$D = \left\{ x \in \mathbb{R}^n \big| Gx \le f \right\}, \tag{2}$$

$G \in \mathbb{R}^{k \times n}, f \in \mathbb{R}^k$. A set $\mathcal{D} \subseteq D$ is a controlled invariant set for system Σ if:

$$x \in \mathcal{D} \implies \exists u \in \mathbb{R} : Ax + Bu \in \mathcal{D}.$$

Given system (1) and the set D (2), we wish to find a controlled invariant subset of D for Σ. We call D the *safe set*.

Next we define the *controlled predecessor* of a polyhedron $D \subset \mathbb{R}^n$ under the system Σ, denoted by $\mathrm{Pre}(X)$:

$$\mathrm{Pre}(D) = \left\{ x \in \mathbb{R}^n \big| \exists u \in \mathbb{R} : Ax + Bu \in D \right\}. \tag{3}$$

Intuitively, $\mathrm{Pre}(D)$ is the set of all states x for which there exists a control input u forcing them into D in one time-step. It follows that for the polyhedron D, $\mathrm{Pre}(D)$ is given by:

$$\mathrm{Pre}(D) = \pi_{\mathbb{R}^n} \left(Q(D) \right),$$

where the auxiliary set $Q(D)$ is:

$$Q(D) = \left\{ (x, u) \in \mathbb{R}^n \times \mathbb{R} \middle| \begin{bmatrix} GA & GB \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \le f \right\},$$

i.e., the set of states x and control inputs u such that x is in D in the next time-step, and $\pi_{\mathbb{R}^n}$ is the natural projection from $\mathbb{R}^n \times \mathbb{R}$ to $\mathbb{R}^n$ mapping $(x, u) \in \mathbb{R}^n \times \mathbb{R}$ to $x \in \mathbb{R}^n$.

To solve Problem 1 an iterative procedure was proposed [1], [2], [7]:

$$\begin{cases} S_0 = X \\ S_{k+1} = S_k \cap \mathrm{Pre}(S_k) \end{cases}, \tag{4}$$

with the terminating condition $S_{k+1} = S_k$. Procedure (4) theoretically works for any discrete-time system, not just for the class of linear systems and convex sets.

In this work, we make the following two *assumptions* on the set D and the discrete-time linear system Σ as part of the problem setup:

1) The set $D \subset \mathbb{R}^n$ is a compact polyhedron;
2) The system Σ, as presented in (1), is controllable.

In order to provide a clean and streamlined mathematical description of the proposed results we work with the Brunovsky canonical form (see [6]) of (1). For any controllable linear system (1), there exist matrices $\Phi \in \mathbb{R}^{n \times n}$ and $\Psi \in \mathbb{R}^{m \times n}$, with Φ invertible, such that the system defined by $(\Phi A \Phi^{-1} - \Phi B \Psi, \Phi B)$ is in Brunovsky normal form. In particular, with the change of coordinates $z = \Phi x \iff \Phi^{-1} z = x$, and the feedback transformation

$u = -\Psi x + v$ we obtain the equivalent system in Brunovsky normal form:

$$z_{k+1} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \ddots & & \vdots \\ \vdots & & \ddots & & 0 \\ \vdots & & & 0 & 1 \\ 0 & 0 & \cdots & \cdots & 0 \end{bmatrix} z_k + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} v_k = A_c z_k + B_c v_k. \tag{5}$$

The safe set in the new coordinates, namely D_c, is of course still a polyhedron in $\mathbb{R}^n$,

$$D_c = \left\{ z \in \mathbb{R}^n \big| G\Phi^{-1} z \le f \right\} = \left\{ z \in \mathbb{R}^n \big| G_c z \le f \right\}. \tag{6}$$

In other words it is an intersection of halfspaces:

$$G_c z \le f \iff g_j^T z \le f_j, j = 1, \cdots, k,$$

where g_j^T is the j-th row of G_c and $f = \begin{bmatrix} f_1 & \cdots & f_k \end{bmatrix}^T$.

Remark 1: In the remainder of the paper, we focus on the case where we have constraints on the states but not on the input. Our results hold in the exact same manner if input constraints are enforced. In this case, we extend the original state space by one dimension, thus obtaining the state $y = (x, u)$, and introduce a new unconstrained input ν governing the evolution of the state u according to $u_{k+1} = \nu_k$.

III. CONTROLLED INVARIANCE IN TWO MOVES

In this section we propose our novel algorithm for computing a controlled invariant set for a discrete-time linear system. We begin by lifting set D_c into a higher dimensional space, where a related MCIS can be computed in closed-form. Then, by projecting the resulting set back to the original space we obtain a controlled invariant subset of D_c. Proofs on correctness and completeness are provided in this section.

A. Lifting the safe set and the system

The first step is to lift the set D_c to a higher dimensional space so as to represent it as a union of hyper-rectangles. Towards this end, we introduce $\lambda \in \mathbb{R}^{kn}$, and lift $D_c \subset \mathbb{R}^n$ to a set $D_c^\ell \subset \mathbb{R}^{n+kn}$ by the following construction:

$$\begin{cases} g_{ji} z_i \le \lambda_{ji}, i = 1, \cdots, n \\ \sum_{i=1}^n \lambda_{ji} \le f_j \end{cases}, j = 1, \ldots, k, \tag{7}$$

where g_{ji} is the entry of G_c in its j-th row and i-th column, and $\lambda = \begin{bmatrix} \lambda_{11} \ldots \lambda_{ji} \ldots \lambda_{kn} \end{bmatrix} \in \mathbb{R}^{kn}$. Then the *lifted* set D_c^ℓ is:

$$D_c^\ell = \left\{ \hat{x} = (z, \lambda) \in \mathbb{R}^{n+kn} \mid \hat{G}_0 \hat{x} \le \hat{f}_0 \right\}, \tag{8}$$

where:
$\hat{G}_0 = \begin{bmatrix} \bar{G}_c & -\mathbb{I}_{kn \times kn} \\ 0_{k \times n} & H \end{bmatrix}$, $\bar{G}_c = \begin{bmatrix} diag(g_1) & \cdots & diag(g_k) \end{bmatrix}^T$, with $diag(g_j)$ a diagonal matrix with the elements of g_j, $H \in \mathbb{R}^{k \times kn}$ such that $H\lambda \le f \iff \sum_i \lambda_{ji} \le f_j, j = 1, \ldots, k$, and $\hat{f}_0 = \begin{bmatrix} 0_{kn \times 1} \\ f \end{bmatrix}$. Note that once we fix λ, (7) defines a collection of hyper-rectangles by restricting z_i to belong to an interval. Hence, we can see D_c^ℓ as the union of all the

hyper-rectangles defined by the coefficients λ_{ji} satisfying the second equation in (7).

Finally, we lift system (5) to Σ^ℓ:

$$\begin{bmatrix} z \\ \lambda \end{bmatrix}_{k+1} = \begin{bmatrix} A_c & 0_{kn\times kn} \\ 0_{n\times n} & I_{kn\times kn} \end{bmatrix} \begin{bmatrix} z \\ \lambda \end{bmatrix}_k + \begin{bmatrix} B_c \\ 0_{kn\times 1} \end{bmatrix} v_k \Longleftrightarrow$$
$$\hat{x}_{k+1} = \hat{A}\hat{x}_k + \hat{B}v_k, \tag{9}$$

where $\hat{x}_k \in \mathbb{R}^{n+kn}$ and $v_k \in \mathbb{R}$. We fix λ to be constant so that rectangles in (7) are preserved by the lifted system, allowing us to compute the MCIS in this higher dimensional space in closed-form.

Figure 1 illustrates the ideas of this subsection: the original set, an octagon, is lifted to a higher dimensional space and represented by a union of different hyper-rectangles depending on the variable λ. For a fixed λ we have an instantiation of the hyper-rectangles. By projecting all these instantiations back to the original domain, one obtains the octagon.

B. Proposed algorithm

The method we propose works with a system in the form (9), and applies the well-known [1], [2], [4], [7] iterative procedure (4), which is proven to yield the maximal control invariant set upon termination. This is summarized in Algorithm 1, which is based on the following high-level idea:

- Given a controllable, discrete-time linear system Σ, and a set D_c, lift $D_c \subset \mathbb{R}^n$ into the set $D_c^\ell \subset \mathbb{R}^{n+kn}$ (8), and system Σ to system Σ^ℓ (9).
- Apply procedure (4) (we prove finite termination in the next subsection).
- Project the set resulting from procedure (4) back to the original vector space.
- The resulting set is controlled invariant.

In the next subsection we prove that procedure (4) terminates in exactly n steps, where n is the dimension of the original system.
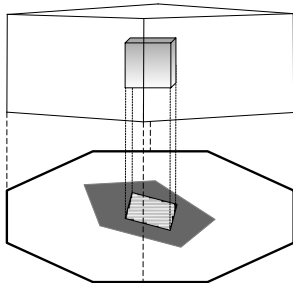

Fig. 1: Illustration of domain lifting. Octagon is the original set D_c. Transparent hyper-rectangle is an instantiation of the hyper-rectangles whose union represents the lifted set D_c^ℓ. Gradient cube is the MCIS in the higher dimension. Light grey rectangle is the controlled invariant set obtained after projection. Dark grey polygon is the exact MCIS.

Algorithm 1: Computation of a controlled invariant subset of a set D.

Data: A set $D_c = \{z \in \mathbb{R}^n \mid G_c z \le f\}$, and a controllable pair (A, B).

Result: A controlled invariant subset of D_c.

Input: G_c, f, A, B

Define:

$\lambda := [\lambda_{11} \dots \lambda_{1n} \dots \lambda_{k1} \dots \lambda_{kn}] \in \mathbb{R}^{kn}$,

$s.t. \sum_{i=1}^n \lambda_{ji} \le f_j, \ j = 1, \cdots, k.$

$D_c^\ell = \left\{ \hat{x} = (z, \lambda) \in \mathbb{R}^{n+kn} \ \middle| \ \hat{G}_0 \hat{x} \le \hat{f}_0 \right\}.$

Init: $S_0 := D_c^\ell$
    while $S_{k+1} \ne S_k$ **do**
        $S_{k+1} := S_k \cap \mathrm{Pre}(S_k)$
    end while
    $\mathcal{D} := \pi_{\mathbb{R}^n}(S_k)$

Return $\mathcal{D}$

C. Finite termination

The result of this subsection was motivated by the similar results in [20], [23], [25] in which finite termination is guaranteed. However, in the higher dimensional space where procedure (4) is applied, the system Σ^ℓ in (9) is not controllable, although controllability is one of the assumptions upon which the aforementioned results rely.

The following Theorem constitutes our main result. In this theorem we make three claims: 1) finite termination, 2) correctness, and 3) completeness of Algorithm 1. While 1) is proven in this subsection, we prove 2) and 3) in separate subsequent theorems.

Theorem 3.1: For any discrete-time linear system (5), and a safe set D_c, for which assumptions 1) and 2) hold, a controlled invariant set $\mathcal{D} \subseteq D_c$ is provided by:

$$\mathcal{D} = \pi_{\mathbb{R}^n}(S_n), \tag{10}$$

where S_n is the MCIS of $D_c^\ell \subset \mathbb{R}^{n+kn}$ for Σ^ℓ, and whose closed-form expression is given in (12).

Proof: Initialize Algorithm 1 with $S_0 = D_c^\ell$. In order to compute $\mathrm{Pre}(S_0)$ using (3), the auxiliary operator $Q(S_0)$ is given by:

$$Q(S_0) = \left\{ (\hat{x}, v) \in \mathbb{R}^{n+kn+1} \ \middle| \ \begin{bmatrix} \hat{G}_0 \hat{A} & \hat{G}_0 \hat{B} \end{bmatrix} \begin{bmatrix} \hat{x} \\ v \end{bmatrix} \le \hat{f}_0 \right\}$$

$$= \left\{ (z, \lambda, v) \in \mathbb{R}^{n+kn+1} \ \middle| \ \begin{bmatrix} \bar{G}_c A_c & -\mathbb{I} & \bar{G}_c B_c \\ 0 & H & 0 \end{bmatrix} \begin{bmatrix} z \\ \lambda \\ v \end{bmatrix} \le \hat{f}_0 \right\}.$$

The inequalities of $Q(S_0)$ can be written as:

$$g_{ji} z_{i+1} - \lambda_{ji} \le 0, \ i = 1, \dots, n-1,$$
$$g_{jn} v - \lambda_{jn} \le 0.$$

To obtain $\mathrm{Pre}(S_0) = \pi_{\mathbb{R}^{n+kn}}(Q(S_0))$ we eliminate the variable v. Towards this end, we define the following sets:

$$P_1 = \{p \mid g_p = g_{jn} > 0\}$$
$$T_1 = \{t \mid g_t = g_{jn} < 0\}$$
$$R_1 = \{r \mid g_r = g_{jn} = 0\}.$$

Using now the Fourier-Motzkin Elimination (FME) method [17] we obtain:

$$\bigwedge_{p \in P_1} \bigwedge_{t \in T_1} \left(g_t \lambda_p - g_p \lambda_t \le 0 \right). \tag{11}$$

By defining a matrix $\Gamma_1 = \begin{bmatrix} 0 & \Gamma_1^\lambda \end{bmatrix}$ such that $\Gamma_1 \hat{x} \le 0$ is equivalent to (11), we can write $\mathrm{Pre}(S_0)$ as:

$$\mathrm{Pre}(S_0) = \left\{ \hat{x} \in \mathbb{R}^{n+kn} \middle| \begin{bmatrix} \left(\hat{G}_0 \hat{A} \right)_{R_1} \\ \Gamma_1 \end{bmatrix} \hat{x} \le \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right\},$$

where the subscript R_1 denotes that we keep only the rows with index belonging to the set R_1. Therefore, at the end of the first step we compute $S_1 = S_0 \cap \mathrm{Pre}(S_0)$:

$$S_1 = \left\{ \hat{x} \in \mathbb{R}^{n+kn} \middle| \begin{bmatrix} \hat{G}_0 \\ \left(\hat{G}_0 \hat{A} \right)_{R_1} \\ \Gamma_1 \end{bmatrix} \hat{x} \le \begin{bmatrix} \hat{f}_0 \\ 0 \\ 0 \end{bmatrix} \right\}.$$

At each step $i \in \mathbb{Z}^+$, all the constraints imposed by the projection of $Q(S_i)$ to obtain $\mathrm{Pre}(S_i)$ are realized by the matrix Γ_i. Notice that for each i, the constraints Γ_{i-1} are included in Γ_i. Utilizing this fact, and applying this procedure iteratively, at the n-th step we obtain:

$$S_n = \left\{ \hat{x} \in \mathbb{R}^{n+kn} \middle| \begin{bmatrix} \hat{G}_0 \\ \hat{G}_1 \\ \vdots \\ \hat{G}_n \\ \Gamma_n \end{bmatrix} \hat{x} \le \begin{bmatrix} \hat{f}_0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \right\}, \tag{12}$$

where $\hat{G}_i = \left[\left(\left(\left((\hat{G}_0 \hat{A})_{R_1} \right) \hat{A} \right)_{R_2} \dots \hat{A} \right)_{R_i} \right]$, $i = 1, \dots, n$ and $\Gamma_n = \begin{bmatrix} 0 & \Gamma_n^\lambda \end{bmatrix}$ such that $\Gamma_n \hat{x} \le 0$ is equivalent to:

$$\bigwedge_{p \in P_n} \bigwedge_{t \in T_n} \left(g_t \lambda_p - g_p \lambda_t \le 0 \right), \tag{13}$$

where index sets P_n, T_n and R_n are:

$$P_n = \{ p \,|\, g_p = g_{jl} > 0, \ l = n, \dots, 1, \ j = 1, \dots, k \}$$
$$T_n = \{ t \,|\, g_t = g_{jl} < 0, \ l = n, \dots, 1, \ j = 1, \dots, k \}$$
$$R_n = \{ r \,|\, g_r = g_{jl} = 0, \ l = n, \dots, 1, \ j = 1, \dots, k \}.$$

If we were to perform one more step, the new constraints to be added depend on the matrix $\hat{G}_n$ multiplied by the dynamics. At each step i, we keep only the rows whose index belongs to R_i. Notice that $\hat{G}_n \hat{B} = 0$ and hence no new constraints involving v are introduced. Moreover, $\hat{G}_n \hat{A} = \hat{G}_n$, and therefore the equality $\hat{G}_{n+1} = \hat{G}_n$ holds. Consequently, no more constraints are added after the n-th step and it follows immediately that $S_{n+1} = S_n$, proving that procedure (4) terminates in exactly n steps. The set S_n is therefore the MCIS of D_c^ℓ. By projecting $S_n \subseteq \mathbb{R}^{n+kn}$ back to $\mathbb{R}^n$, we obtain:

$$\mathcal{D} = \pi_{\mathbb{R}^n}(S_n).$$

∎

Remark 2: Although we only discuss single-input systems, it is easy to see that the same argument works for controllable multiple-input systems since they can be decomposed to several decoupled single-input systems.

D. Proof of correctness

In this subsection we prove that the set computed by Algorithm 1 is indeed a *controlled invariant* set.

Theorem 3.2: For any discrete-time linear system and set D that satisfy assumptions 1) and 2), if Algorithm 1 returns a non-empty set $\mathcal{D}$, then $\mathcal{D}$ is a controlled invariant subset of D for Σ.

Proof: For any $x \in \mathcal{D}$, there exists a $\hat{x} \in S_n$ such that $\pi_{\mathbb{R}^n}(\hat{x}) = x$. Since S_n is controlled invariant, it follows that $\hat{A}\hat{x} + \hat{B}v \in S_n$. Consequently, $\pi_{\mathbb{R}^n}(\hat{x}) \in \mathcal{D}$. The proof is now finished by noting that $Ax + Bv = \pi_{\mathbb{R}^n}(\hat{A}\hat{x} + \hat{B}v) \in D$. ∎

E. Proof of Completeness

Here we prove that if the MCIS of a compact polyhedron D is non-empty, then Algorithm 1 will always find a non-empty controlled invariant subset of D.

Theorem 3.3: For any discrete-time linear system and set D that satisfy assumptions 1) and 2), if the MCIS of D is non-empty, then Algorithm 1 returns a non-empty set $\mathcal{D}$ which is a controlled invariant subset of D.

Proof: Due to space limitations, we only provide a *sketch* of the proof.

For a system Σ given by (1) assume that $\tilde{D}$ is the non-empty MCIS of a compact, convex set $D \subset \mathbb{R}^n$. Then there exists a stationary control law $\xi(x)$ for every $x \in \tilde{D}$ such that $Ax + B\xi(x) \in \tilde{D}$ [2, Cor.1]. Now let $\mathcal{K}$ be the set-valued function that maps a point $x \in \tilde{D}$ to the set of all admissible control inputs u such that $Ax + Bu \in \tilde{D}$. Formally:

$$\mathcal{K}(x) = \begin{cases} \{ u \in \mathbb{R} \,|\, Ax + Bu \in \tilde{D} \}, & x \in \tilde{D} \\ \varnothing, & x \notin \tilde{D} \end{cases}. \tag{14}$$

Our goal is to prove the set $\tilde{D}$ admits a fixed point with respect to $Ax + B\xi(x)$. Towards this we use the following two results from the literature:

Theorem 3.4 (Thm. 3.2 [16]): If A is a paracompact space, B a Banach space and $h : A \to 2^B$ a lower semi-continuous mapping with nonempty closed convex values, then h admits a continuous single-valued selection.

Proposition 3.5 (Prop. 3.8 [11]): If h is a C-convex mapping, and efficient mapping at a point x_0 of its domain, then h is also lower semicontinuous at x_0.

The following can be proven in a straightforward manner: (1) $\mathcal{K}$ is closed convex valued, (2) $\mathcal{K}$ is C-convex (Sec. 2 [11]), and (3) $\mathcal{K}$ is efficient (Def. 3.5 [11]).

Consequently, by 2), 3) and [11, Prop. 3.8] $\mathcal{K}$ is a lower semicontinuous mapping. Looking at [16, Thm. 3.2], the space $A = \tilde{D} \subseteq \mathbb{R}^n$ is paracompact since every metric space is paracompact [22], the space $B = \mathbb{R}$, i.e., the one-dimensional Eucledean space, is a Banach space, and by 1) $\mathcal{K}$ has closed convex values. Therefore, $\mathcal{K}$ admits a continuous singled-valued selection, $\kappa : \tilde{D} \to \mathbb{R}$.

As a result, one can now write the system as $Ax + Bu = f(x, u) = f(x, \kappa(x))$, and f is now continuous with respect to x. By Brouwer's fixed-point theorem [5], the compact and convex set $\tilde{D}$ has a fixed-point $\tilde{x}$ with respect to the continuous mapping f.

If we write $\{\tilde{x}\} \subset \tilde{D}$ as a "rectangle" in $\mathbb{R}^n$, namely $D_{rec} = \{ x \in \mathbb{R}^n \,|\, \tilde{x}_i \le x_i \le \tilde{x}_i, \ i = 1, \dots, n \}$, transform it

to the Brunovsky coordinates, and then lift it from $\mathbb{R}^n$ to $\mathbb{R}^{n+kn}$, we obtain a non-empty set X_{rec} which is controlled invariant in $\mathbb{R}^{n+kn}$. Thus, there exists a *non-empty* MCIS $\tilde{X} \subset \mathbb{R}^{n+kn}$, since $X_{rec} \subseteq \tilde{X}$, which in turn implies existence of a point $\tilde{x}_0 \in \tilde{X}$, and a control sequence $\tilde{v}(\tilde{x}_0) = \{\tilde{v}_0(\tilde{x}_0), \tilde{v}_1(\tilde{x}_1), \dots\}$, such that the trajectory starting at $\tilde{x}_0$ under $\tilde{v}(\tilde{x}_0)$ is still in $\tilde{X}$ after n steps. This implies $\tilde{X} \subseteq S_n \Rightarrow S_n \neq \varnothing$. Finally, by virtue of (10): $\mathcal{D} = \pi_{\mathbb{R}^n}(S_n) \neq \varnothing$, concluding the proof. ∎

IV. ILLUSTRATIVE EXAMPLES AND COMPUTATIONAL EVALUATION

We begin this section by presenting two examples, one in $\mathbb{R}^2$ with constrained input, and one in $\mathbb{R}^3$ with unconstrained input, to illustrate how our approach can handle systems with peculiarly shaped convex domains. Later in this section, we perform a computational evaluation showing that this approach can handle larger systems.

Example 1: Consider the following system in $\mathbb{R}^2$:

$$x_{k+1} = \begin{bmatrix} 1.5 & 1 \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} 0.5 \\ 0.25 \end{bmatrix} u_k,$$

$D = \{x \in \mathbb{R}^2 | Gx \leq f\}$, with G and f such that D is the intersection of 6 halfspaces in $\mathbb{R}^2$, $u \in [-2, 2]$ (Fig. 2a), and $u \in [-1, 1]$ (Fig. 2b).

In Figure 2a, the union of all sets is D, the union of white and light gray sets is the MCIS of D, the result of Algorithm 1, $\mathcal{D}$, is in white, and the set difference between the MCIS and $\mathcal{D}$ is in light gray. In Figure 2b, one can see how both controlled invariant sets (MCIS and $\mathcal{D}$) shrink as the constraints on control become tighter.

Example 2: Consider the following system in $\mathbb{R}^3$:

$$x_{k+1} = \begin{bmatrix} 0 & 1 & -2 \\ 3 & -4 & 5 \\ -6 & 7 & 8 \end{bmatrix} x_k + \begin{bmatrix} -1 \\ 2 \\ 4 \end{bmatrix} u_k,$$

$D = \{x \in \mathbb{R}^3 | Gx \leq f\}$, with G and f such that D is the intersection of 12 halfspaces in $\mathbb{R}^3$, and $u \in \mathbb{R}$.

In Figure 2c, using again the same color notation for sets, D is shown. One can see in detail, in Figures 2d, 2e, and 2f, how the MCIS of D (union of white and light gray sets) compares with the result of Algorithm 1, $\mathcal{D}$ (white), using slices along each axis.

In both previous toy examples, the point was to show how our approach handles cases with peculiar polyhedra. However this is not the highlight of this approach. Contrary to other approaches, the algorithm in this paper is able to handle higher dimensional systems as well.

Remark 3: Note that in terms of performance of Algorithm 1 we deal only with the single-input case. This performance generalizes to the multiple-input case, as it is measured in terms of the maximum controllability index μ of a system. A controllable system with m inputs can be decomposed to m single-input decoupled systems. The largest of them has dimension μ, which dominates in terms of the computational cost.
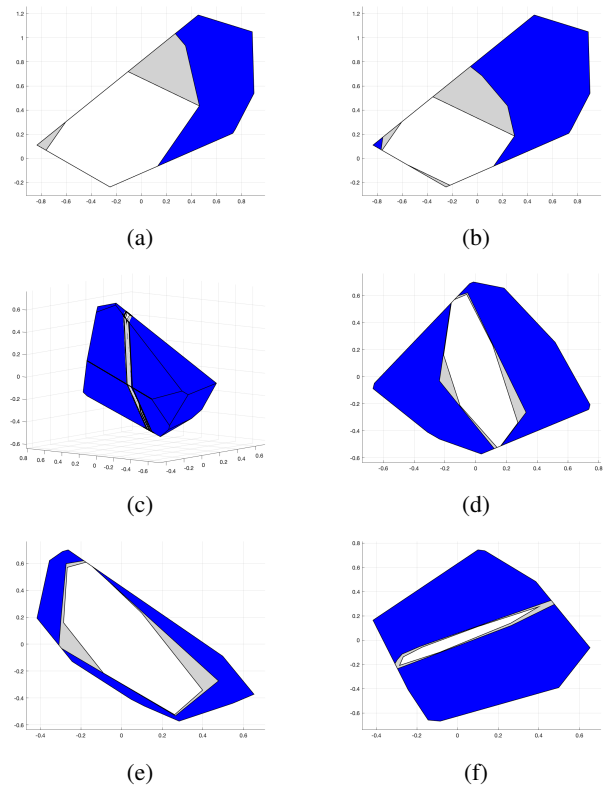

(a)   (b)   (c)   (d)   (e)   (f)

Fig. 2: (a) Sets of Example 1 with $u \in [-2, 2]$. (b) Sets of Example 1 with $u \in [-1, 1]$. (c) Sets of Example 2. (d) 2D slice of (c) in z-axis. (e) 2D slice of (c) in x-axis (e) 2D slice of (c) in y-axis

Legend: Original set D (blue); Exact MCIS of D (light gray); Controlled invariant set $\mathcal{D}$ from Algorithm 1 (white).

Consequently, one computes a controlled invariant set of an arbitrarily high dimensional system with n states, m inputs, and maximum controllability index μ by computing in parallel controlled invariant sets of m decoupled subsystems. This holds since once we lift the set, it becomes the product of hyper-rectangles and thus the lifted set also decomposes. Simulations have shown that this method is computationally not significantly more expensive than having a single system of dimension μ.

Therefore, in Table I we present for reference the mean execution times for systems with dimensions $n = 2$ to $n = 10$, obtained over 20 different trials each with a different randomly generated compact polyhedron with $2n$ constraints. Since it is equivalent to work with a system in Brunovsky normal form, for our simulations we assumed that each system is in this form. Bringing any system to this form is computationally cheap compared to the obtaining a controlled invariant set, and hence does not effectively increase the reported times.

As a benchmark, we tried to compute the MCIS using the `invariantSet()` function of the Multi-Parametric Toolbox (MPT) [10] for the same systems. The corresponding times are reported in Table I. Note that the reported times for MPT involve only 5 iterations of the approach used. This

6253

TABLE I: Mean execution times of Algorithm 1 and MPT3 for an n-dimensional system and a randomly generated polyhedron with $2n$ constraints. NA indicates aborting the simulation if not concluded in 4 hours.

	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n=10$
Algorithm 1	0.05	0.19	2.06	9.24	42.37	73.47	332.29	1942	~ 15000
MPT 5 iterations	0.16	0.30	0.93	24.27	1384	NA	NA	NA	NA

means that for $n > 3$, in the vast majority of cases, the computation did not converge and the resulting sets are not actually controlled invariant. More specifically, notice how quickly the computational time of MPT3 explodes, and in particular for $n > 6$ the iterations were not concluded after four hours and the simulation was thus aborted.

Implementing the projection from $\mathbb{R}^{n+kn}$ back to $\mathbb{R}^n$, we exploit the structure of the MCIS computed in closed-form, using a technique inspired by [21]. The authors there, pick at each FME step the variable that yields the least growth, i.e., number of new inequalities introduced, and do not allow the system of inequalities to grow beyond the original size. If it does, they approximate the projection. However we utilized exact projection instead of approximating it, and also allowed the system to grow slightly beyond the original size.

For reference[1], all the simulations were conducted with an iMac (Late 2012), with 4 cores @ 3.4 GHz Intel Core i7 Processor, and 32 GB 1600 MHz DDR3 RAM.

V. CONCLUSION

In this paper, we presented a novel algorithm for computing invariant sets of discrete-time linear systems in two moves: 1) we lift the problem to a higher dimensional space where the controlled invariant set is computed in closed-form; 2) we project the resulting set back to the original domain. This algorithm, which does not rely on iterative computations, has advantages over other methods that trade off computational efficiency and accuracy. First, it is complete as it is guaranteed to compute a non-empty controlled invariant set if the MCIS is not empty. Second, although it is not guaranteed to compute the MCIS, it computes a set that is guaranteed to be controlled invariant. Another advantage of the proposed method is the ability to handle larger systems measured in terms of their controllability index. Finally, we believe the performance of the proposed method can be further improved by exploiting the specific structure of the MCIS computed in closed-form in the higher dimensional space, which is the focus of our current research.

REFERENCES

[1] D. P. Bertsekas and I. B. Rhodes, "On the minimax reachability of target sets and target tubes," *Automatica*, vol. 7, no. 2, pp. 233–247, Mar. 1971.

[2] D. Bertsekas, "Infinite time reachability of state-space regions by using feedback control," *Automatic Control, IEEE Transactions on*, vol. AC-17, pp. 604 – 613, 11 1972.

[3] F. Blanchini, "Survey paper: Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, Nov. 1999.

[4] F. Blanchini and S. Miani, *Set–Theoretic Methods in Control*, ser. Systems & Control: Foundations & Applications. Boston, Basel, Berlin: Birkhauser, 2008.

[5] L. Brouwer, "Über Abbildung von Mannigfaltigkeiten," *Mathematische Annalen*, vol. 71, pp. 97 – 115, 1912.

[6] P. Brunovský, "A classification of linear controllable systems," *Kybernetika*, vol. 6, pp. 173–188, 1970.

[7] J. Duncan Glover and F. C. Schweppe, "Control of linear dynamic systems with set constrained disturbances," *Automatic Control, IEEE Transactions on*, vol. 16, pp. 411 – 423, 11 1971.

[8] M. Fiacchini and M. Alamir, "Computing control invariant sets is easy," *CoRR*, vol. abs/1708.04797, 2017.

[9] J. H. Gillula, S. Kaynama, and C. J. Tomlin, "Sampling-based approximation of the viability kernel for high-dimensional linear sampled-data systems," in *Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '14. New York, NY, USA: ACM, 2014, pp. 173–182.

[10] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/ mpt.

[11] F. Heyde and C. Schrage, "Continuity concepts for set-valued functions and a fundamental duality formula for set-valued optimization," *Journal of Mathematical Analysis and Applications*, vol. 397, no. 2, pp. 772 – 784, 2013.

[12] S. Kaynama, J. Maidens, M. Oishi, I. M. Mitchell, and G. A. Dumont, "Computing the viability kernel using maximal reachable sets," in *Proceedings of the 15th ACM International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '12. New York, NY, USA: ACM, 2012, pp. 55–64.

[13] E. C. Kerrigan and J. M. Maciejowski, "Invariant sets for constrained nonlinear discrete-time systems with application to feasibility in model predictive control," in *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No.00CH37187)*, vol. 5, Dec 2000, pp. 4951–4956 vol.5.

[14] M. Lazar, "Model predictive control of hybrid systems : stability and robustness," Ph.D. dissertation, Department of Electrical Engineering, 2006.

[15] B. Legat, P. Tabuada, and R. M. Jungers, "Computing controlled invariant sets for hybrid systems with applications to model-predictive control," in *6th IFAC Conference on Analysis and Design of Hybrid Systems, ADHS 2018, Oxford, UK, July 11-13, 2018*, 2018, pp. 193–198.

[16] E. Michael, "Continuous Selections. I," *Annals of Mathematics*, vol. 63, no. 2, pp. 361–382, 1956.

[17] T. S. Motzkin, *Beiträge zur Theorie der linearen Ungleichungen*. Azriel Press, 1936.

[18] M. Rungger, M. Mazo, and P. Tabuada, "Scaling up controller synthesis for linear systems and safety specifications," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, Dec 2012, pp. 7638–7643.

[19] M. Rungger and P. Tabuada, "Computing robust controlled invariant sets of linear systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 7, pp. 3665–3670, July 2017.

[20] M. Rungger, M. Mazo, Jr., and P. Tabuada, "Specification-guided controller synthesis for linear systems and safe linear-time temporal logic," in *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '13. New York, NY, USA: ACM, 2013, pp. 333–342.

[21] A. Simon and A. King, "Exploiting sparsity in polyhedral analysis," in *Proceedings of the 12th International Conference on Static Analysis*, ser. SAS'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 336–351.

[22] A. H. Stone, "Paracompactness and product spaces," *Bulletin of the American Mathematical Society*, vol. 54, no. 10, pp. 977–982, 10 1948.

[23] P. Tabuada and G. J. Pappas, "Linear time logic control of discrete-time linear systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 12, pp. 1862–1877, Dec 2006.

[24] P. Trodden, "A one-step approach to computing a polytopic robust positively invariant set," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 4100–4105, Dec 2016.

[25] R. Vidal, S. Schaffert, J. Lygeros, and S. Sastry, "Controlled invariance of discrete time systems," in *Proceedings of the Third International Workshop on Hybrid Systems: Computation and Control*, ser. HSCC '00. London, UK, UK: Springer-Verlag, 2000, pp. 437–450.

[1]The code for both examples of this section, as well as a code that generates random compact polyhedra, and the code to our approach are freely available online at https://github.com/janis10/cis2m.